ISSN 1561-5359. Штучний інтелект, 2016, № 2

УДК 004.93(075.8)

В. М. Терещенко, Я. В. Терещенко
Київський національний університет імені Тараса Шевченка, Україна
вул. Володимирська 60, м. Київ, 01033

ОДИН ПІДХІД ДО РОЗПІЗНАВАННЯ ГЕОМЕТРИЧНИХ ОБ'ЄКТІВ У ЗАДАЧАХ КОМП'ЮТЕРНОГО ЗОРУ

V. M. Tereshchenko , Y. V. Tereshchenko
Taras Shevchenko National University of Kyiv, Ukraine
60 Volodymyrska Street, City of Kyiv, Ukraine, 01033

ONE APPROACH TO RECOGNIZING GEOMETRIC OBJECTS IN COMPUTER VISION PROBLEMS

Стаття присвячена розробці одного з підходів до розв'язання задач комп'ютерного зору і, зокрема, розпізнавання геометричних об'єктів. Цей підхід заснований на використанні ефективних алгоритмів обчислювальної геометрії і, зокрема, методу монотонних ланцюгів. Також у роботі, для розпізнавання многокутників, застосовується персептрон Розенблата.

Ключові слова: штучний інтелект, комп'ютерний зір, розпізнавання, геометричний об'єкт, алгоритми обчислювальної геометрії, метод ланцюгів.

The paper is devoted to the use of one approach to addressing the various types of problems of pattern recognition. This approach is based on use of effective algorithms for computational geometry, and particular, the method of monotone chains. Also, in the paper we use Rosenblatt perceptron for recognition polygons.

Keywords: artificial intelligence, computer vision, pattern recognition, geometric object, computational geometry algorithms, the method of monotone chains.

Вступ

Існують такі напрямки штучного інтелекту, які мають застосування у вузько спеціалізованих галузях, де необхідні автоматизовані системи розпізнавання. Прикладом таких систем є автоматизовані системи розпізнавання ситуацій, які зчитують зображення із цифрових пристроїв (відеокамери чи фотоапарату), розпізнають їх та, використовуючи результати роботи попереднього кроку, на основі заданих правил, виконують відповідні дії. Ефективність роботи систем такого типу цілком залежить від того, наскільки якісно буде розпізнана вхідна інформація [1]. У роботі пропонується стратегія, яка пов'язана із визначенням класів об'єктів за їх характеристиками (проблема ідентифікації об'єктів). Розглядається спрощена модель, у якій множина класів об'єктів, за допомогою узагальненого методу редукції (УМР) [6], зводиться до множини класів многокутників на площині, які в свою чергу описуються упорядкованими множинами ребер(відрізків). Такий частковий випадок легко зводиться до загальної проблеми ідентифікації об'єктів як на площині, так і в просторі.

Постановка проблеми

Розглядається задача розпізнавання образів на деякій площині. У геометричній інтерпретації її можна сформулювати наступним чином.

Задача 1. На площині задана деяка скінченна множина відрізків $A=\{a|\ a=(x_1,y_1, x_2,y_2)\}$, кожен елемент якої задається чотирма координатами, де (x_1,y_1) та (x_2,y_2) – координати початку та кінця кожного з відрізків. Необхідно на заданій множині A розпізнати многокутники і визначити, якому класу вони належать: *прості, зіркові чи опуклі.*

Введемо деякі означення, поняття та припущення.

Припущення 1. Припустимо, що множина A не містить відрізків, що лежать на одній прямій і мають спільні точки. У задачі такі відрізки можна замінити одним, який містить у собі обидва відрізки. Окрім того $(x_1,y_1, x_2,y_2)= (x_2,y_2, x_1,y_1)$.

Означення 1. Точки перетину двох або більше прямих назвемо *вершинами.*

Тобто, множина A породжує множину $B=\{b|\ b=(x^b, y^b)\}$, де

$$\forall b=(x^b, y^b)\in B\ \exists a_1=(x_{1,1},y_{1,1}, x_{1,2},y_{1,2})\in A,\quad \exists a_2=(x_{2,1},y_{2,1},x_{2,2},y_{2,2})\in A,\ \exists t_1,t2\in[0,1]:$$

$$x^b=x_{1,1}+(x_{1,2}-x_{1,1})t_1 = x_{2,1}+(x_{2,2}-x_{2,1})t_2 \text{ та}$$

$$y^b=y_{1,1}+(y_{1,2}-y_{1,1})t_1 = y_{2,1}+(y_{2,2}-y_{2,1})t_2.$$

Введемо третю множину C ребер, що з'єднують вершини множини B. Тобто, це така множина, породжена множинами B та C, для якої виконується:

$\forall c=(x^c{}_1, y^c{}_1, x^c{}_2, y^c{}_2)\in C\ \exists b_1,b_2\in B,\ \exists a=(x_1,y_1, x_2,y_2)\in A:$

$$b_1=(x^c{}_1, y^c{}_1),\ b_2=(x^c{}_2, y^c{}_2),\ b1,b2\in L,$$

де $L=\{(x,y)\in \mathbf{R}^2|\ x=x_1+(x_2-x_1)t,\ \ y=y_1+(y_2-y_1)t,\ t\in[0,1]\}$ та $\forall b_3\in B:\ b_3\neq b_1,\ b_3\neq b_2 \Rightarrow b_3\notin L$.

Зрозуміло, що остання умова виключає деяку невизначеність, що може з'явитися, якщо декілька вершин породжені перетином одного відрізка з декількома.

Припущення 2. Припустимо $(x^c{}_1, y^c{}_1, x^c{}_2, y^c{}_2)= (x^c{}_2, y^c{}_2, x^c{}_1, y^c{}_1)$.

Означення 2. Під *ланцюгом* з кінцями b_1, b_n будемо розуміти таку послідовність різних точок$(b_1,\ldots,b_n)$ з множини B, для якої виконується:

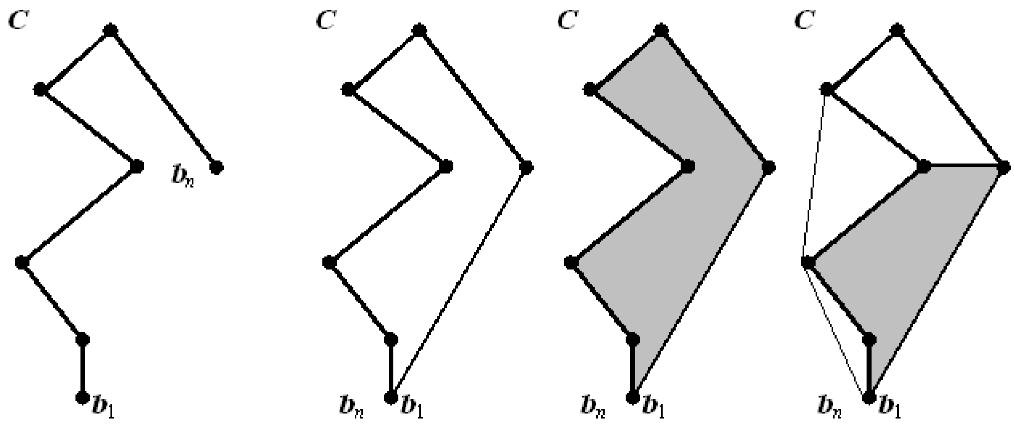
$\forall s=1,\ldots, n-1\ \ \exists c=(x^c{}_1, y^c{}_1, x^c{}_2, y^c{}_2,)\in C:\ b_s=(x^c{}_1, y^c{}_1),\ b_{s+1}=(x^c{}_2, y^c{}_2)$ або $b_{s+1}=(x^c{}_1, y^c{}_1),\ b_s=(x^c{}_2, y^c{}_2)$.

Означення 3. Під *замкненим ланцюгом* будемо називати такий ланцюг, для якого $\exists c=(x^c{}_1, y^c{}_1, x^c{}_2, y^c{}_2,)\in C:\ b_n=(x^c{}_1, y^c{}_1),\ b_1=(x^c{}_2, y^c{}_2)$ або $b_1=(x^c{}_1, y^c{}_1),\quad b_n=(x^c{}_2, y^c{}_2)$.

Згідно з відомою теоремою Жордана замкнена ламана на площині описує многокутник, який розбиває площину на дві області: внутрішню – скінченну та зовнішню - напівскінченну. Внутрішню область назвемо *площею***.**

Означення 4. Під *фігурою* будемо розуміти такий замкнений ланцюг$(b_1,\ldots,b_n)$, для якого не існує жодного ланцюга з кінцями b_s, $b_m\in(b_1,\ldots,b_n)$ такого, щоб усі його точки, крім кінців, належали б площі, обмеженій замкненим ланцюгом$(b_1,\ldots,b_n)$.

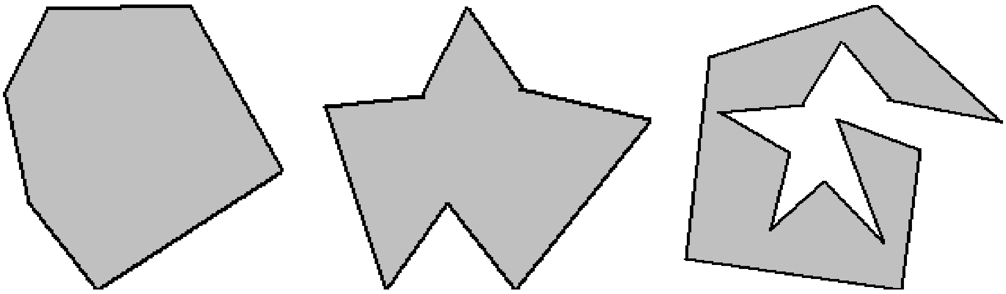
Приклад 1. На рис.1 наведено приклади, які пояснюють геометричний зміст означень.



а) б) в) г)

Рис.1. а) ланцюг, б) замкнений ланцюг, в) площа, г) фігура

Обмежимось розглядом таких відомих в обчислювальній геометрії многокутників, як опуклий, зірковий та простий [7] (рис. 2).



а) б) в)

Рис. 2. а) опуклий многокутник; б) зірковий многокутник; в) простий многокутник

Спираючись на ці означення, можна більш формалізувати поставлену задачу (задача 1).

Задача 2. Нехай на площині задана деяка скінченна множина відрізків А. Аналізуючи цю множину, необхідно породити з неї множин В та С і розпізнати усі фігури, які вони породжують. А також для кожної з фігур необхідно провести їх класифікацію.

Аналіз останніх досліджень і публікацій

На сьогоднішній день мало досліджений клас задач стосовно побудови ефективної стратегії їх розв'язання. У той же час, деякі напрямки штучного інтелекту, які використовують розв'язки таких задач мають значні досягнення. В основному, це ті напрямки, для яких існує чітка математична теорія й на результати яких є попит щодо застосування. Зокрема, одним з таких напрямків є розпізнавання геометричних образів визначеного типу. Одним з найвідоміших прикладів є розпізнавання графічних матричних образів за зразком (наприклад, розпізнавання відсканованих друкованих текстів) [1, 2, 3].

У роботі розглядається застосування одного з підходів до розв'язування різних типів задач розпізнавання образів, який базується на використанні ефективних алгоритмів обчислювальної геометрії. Зокрема, в роботі запропонована модель, яка дозволяє звести класичну задачу розпізнавання образів до задачі розпізнавання геометричних об'єктів простої форми, зокрема множини відрізків, для розв'язання якої існують надшвидкі алгоритми розпізнавання та геометричного пошуку [4 -23]. Ця модель дозволяє розробити швидкий алгоритм розв'язання поставленої задачі, а також визначити характерні риси зображення та має широке коло застосувань У задачах розпізнавання, класифікації та прогнозування ринкових котирувань [5].

1. Опис алгоритмів розв'язання поставленої задачі

Розв'язання цієї задачі проходить у декілька етапів, а тому й розгляд алгоритмів розв'язання доцільно було б зробити поетапно.

1.1. Алгоритм пошуку вершин (точок перетину прямих)

Відомо, що для кожного відрізку(x_1,y_1,x_2,y_2) можна визначити такий прямокутник, сторони якого паралельні осям координат і відрізок(x_1,y_1,x_2,y_2) є в ньому діагоналлю.

1. Формуємо два масиви. У перший записуємо вертикальні сторони отриманих прямокутників (фактично це є координати x_1 та x_2). У другий записуємо горизонтальні сторони отриманих прямокутників (фактично це є координати y_1 та y_2).

2. Сортуємо обидва масиви швидким сортуванням (логарифмічним), але залишаємо зв'язки кожного відрізку зі сторонами прямокутника, що до нього відноситься.
3. Циклічно переглядається кожен прямокутник.
4. Бінарним пошуком за час *O* (log*N*) знаходимо у першому масиві місце лівої та правої сторін прямокутника (у другому, відповідно, верхньої на нижньої). Це - пара чисел, які обмежують підмасив сторін, які належать тим прямокутникам, діагоналі яких перевірятимуться на наступних кроках (точніше, одну діагональ, що співпадає з ребром, по якому будувався прямокутник).
5. Із двох масивів сформуємо два підмасиви, які об'єднаються в один масив.
6. Визначимо ребро, що відповідає многокутнику, який переглядається на даному кроці й перевіряється на перетин з тією множиною ребер, яка була отримана після попередньої операції вибору з двох масивів.
7. Сформовану таким чином множину вершин сортуємо у лексикографічному порядку і заносимо у список ***B***. Але сформована множина створюється крок за кроком й тому перед тим, як включити нову вершину до списку ***B***, необхідно перевірити на наявність її в даній множині.
8. Формуємо множину ***C***, до якої включаємо усі ребра, що лежать між парами сусідніх точок перетину, отриманих після сортування.

1.2. Алгоритм вилучення вершин першого ступеня, та похідних вершин першого ступеня

Означення 5. *Вершиною першого ступеня* будемо називати вершину, яка зв'язана максимум з одним ребром.

Означення 6. *Похідною вершиною першого ступеня* будемо називати таку вершину, яка не є вершиною першого ступеня, але стає такою, якщо з графу вилучити усі вершини першого ступеня та зв'язані з ними ребра.

Похідні вершини також бувають різного ступеня, так як таке означення передбачає ітеративний процес. Тобто, на кожному кроці цього процесу перевіряємо чи є вершини першого ступеня та вилучаємо їх. Якщо на певному кроці вершина першого ступеня відсутня, то процес завершується. Неважко перевірити, що вилучення усіх вершин першого ступеня та похідних вершин першого ступеня зробить з вихідного графу або порожній, або такий, де кожне ребро належить мінімум одній фігурі. Для вилучення вершин таких типів будемо використовувати наступний алгоритм.

Алгоритм

1. Переглядаємо поступово всю множину вершин.
2. Якщо обрана вершина є вершиною першого ступеня, то запам'ятовуємо вершину, яка можливо з нею зв'язана (якщо вона існує).
3. Вилучаємо вершину першого ступеня й зв'язані з нею ребра.
4. Переглядаємо вершину, яку запам'ятали.
5. Якщо на якомусь кроці вершина, яку ми запам'ятали не є першого ступеня, то зупиняємось та повертаємось до перегляду множини вершин, враховуючи те, що з множини вершин деякі були видалені під час перегляду по дереву.

1.3. Метод ланцюгів

1.3.1 Підготовка до методу ланцюгів

На попередньому кроці з матриць ***B*** та ***C*** були одержані модифіковані матриці ***B'*** та ***C'***, які містять тільки ті вершин та ребра, які належать хоча б одній фігурі. Тепер можна переходити до використання методу ланцюгів, детально описаного в роботах [8, 7]. Основна ідея цього методу полягає в тому, що на заданому графі G (V, E) виділяється лінійно впорядкована множина ланцюгів, монотонних відносно деякої

прямої (осі *OY*), використовуючи алгоритм балансування ваг. Слід зауважити, що ці ланцюги можуть мати спільними необмежене число точок, але ж завжди існує така точка, яка належить лише одному з ланцюгів. Потім за допомогою таких ланцюгів можна буде легко виокремити фігури. Підготовка до методу ланцюгів проходить у кілька етапів.

1. Спочатку сортується лексографічно масив вершин (пріоритет по координаті *y*).
2. Знаходиться найнижча ліва та найвища права вершини. Це будуть початковий та скінченний вузли, відповідно.
3. Переглядаються вершини у порядку, в якому вони відсортовані (від початкового до кінцевого вузла) й сортуються ребра за полярним кутом.
4. Усі відсортовані ребра орієнтуємо від нижчої вершини до вищої. Неважко зрозуміти, що інших ребер, що входять у дану вершину, при подальшому перегляді не утвориться.
5. Додання допоміжних ребер. Враховуючи крок регуляризації заданого графа, переглядаємо усі вершини, крім початкової та скінченної. Якщо вершина не має вхідних ребер, то додаємо ребро, що виходить з найближчої меншої вершини (у лексикографічному порядку) й входить у вершину, що переглядаємо. Аналогічно робимо, якщо вершина не має вихідних ребер. Тільки додаткове ребро проводимо у більшу вершину. Усі додаткові ребра запам'ятовуємо. Вибір найближчої вершини не є випадковим. Тим самим гарантується, що не відбудеться перетину ребер на графі.
6. Балансування ваг [7].

Метод ланцюгів запропонований Лі та Препаратою [7] для розв'язання задачі локалізації точки на планарному розбитті в обчислювальній геометрії. У нашому випадку, метод працює лише з графом, що підготовлений на кроці 3. Пошук ведеться звичайним методом пошуку у глибину.

Алгоритм

1. Починаємо з початкового вузла й просуваємось угору до кінцевого.
2. Якщо ми знаходимось у певній вершині на якомусь кроці, то обираємо саме ліве ребро, що виходить з цієї вершини та має ненульову вагу.
3. Якщо такого ребра не існує, й ми знаходимось у вихідному вузлі, то запам'ятовуємо обраний ланцюг й зменшуємо ваги усіх ребер, що до нього входять, на 1.
4. Після цього повертаємось на крок назад.
5. Якщо такого ребра не існує й ми знаходимось у вхідному вузлі, то припиняємо пошуки, бо усі ланцюги знайдено.
6. У випадку, коли не існує вихідного ребра з ненульовою вагою й ми не знаходимось ні у початковому, ні у скінченному вузлі, то необхідно повернутись на крок назад й виконати перевірку знову.

Цей алгоритм, як вже було зазначено, дозволяє виокремити упорядковані монотонні ланцюги. Суміжні ланцюги мають принаймні дві спільні точки. Тому два суміжних ланцюги обов'язково утворюють фігуру.

1.4 Виділення фігур та об'єднання псевдофігур

Ця задача стає тривіальною при наявності монотонних ланцюгів. Для цього потрібно послідовно переглядати усі пари суміжних ланцюгів. При кожному перегляді виокремлюємо з пари суміжних ланцюгів усі такі ланцюги, які мають спільні лише початок і кінець. Зрозуміло, що попарне об'єднання таких ланцюгів дає нам шукані фігури.

Означення 7. *Псевдофігурою* будемо називати таку фігуру, яка містить додані на кроці 3 додаткові ребра.

Алгоритм

1. Переглядаємо увесь масив фігур, отриманих на попередньому кроці.
2. Якщо фігур із доданими ребрами немає, то пошук припиняється.
3. Якщо така фігура знайдена, то фіксуємо одне з її доданих ребер і переглядаємо масив знову.
4. Якщо таке фіксоване ребро більше ніде не зустрічається, то вилучаємо розглядувану фігуру, й продовжуємо перегляд масиву фігур.
5. Якщо таке ребро зустрічається у тій самій фігурі двічі, то виокремлюємо два ланцюги, що розділяються по цьому ребру (кратному). При цьому псевдофігура розщеплюється на дві фігури (у загальному випадку вони також можуть бути псевдофігурами).
6. Продовжуємо перегляд масиву фігур.
7. Якщо таке ребро зустрічається в іншій фігурі, то вилучаємо з обох фігур псевдоребра й об'єднуємо ці фігури у місці розриву в одну фігуру (у загальному випадку вони також можуть бути псевдофігурами).

Після кількох таких кроків усі псевдоребра зникають з фігур.

2. Розпізнавання фігур

Розпізнавання фігур включає в себе розпізнавання кількості кутів, опуклості многокутників та, власне, класифікацію самих геометричних образів. Хоча, на перший погляд, здається, що розпізнавання кількості кутів це - тривіальна задача, треба враховувати випадки розгорнутих кутів і способи їх обробки. Сам же алгоритм виокремлення таких кутів є тривіальним. Необхідно лише перевірити усі пари суміжних ребер. Якщо визначник з координатами векторів, що лежать на ребрах, дає число, близьке до нуля (з визначеним ступенем точності), то можна вважати, що такі суміжні ребра утворюють розгорнутий кут і його непотрібно враховувати. Розв'язання задачі розпізнавання опуклості теж не викликає проблем. Один і з алгоритмів може бути таким.

2.1 Алгоритм розпізнавання опуклості многокутників.

Спочатку орієнтуємо усі ребра проти годинникової стрілки, а потім переглядаємо усі пари суміжних ребер. Ці ребра визначають трійку вершин, які утворюють трикутник (можливо вироджений). Знаходимо центроїд цього трикутника та визначаємо два вектори, що з'єднують центроїд з початком та кінцем будь-якого вектора з пари суміжних, й у цьому ж порядку записуємо ці вектори у верхній та нижній рядок визначника відповідної матриці. Якщо визначник матриці стає від'ємним хоча б для однієї з пар суміжних векторів, то перегляд таких пар припиняємо й вважаємо, що многокутник, який ми перевіряємо не є опуклим. У іншому випадку многокутник є опуклим. Перейдемо до розпізнавання многокутників і розглянемо алгоритм, який використовує персептрон Розенблата [9], на прикладі задачі розпізнавання зіркового многокутника.

2.2 Алгоритм розпізнавання зіркових многокутників.

Для розпізнавання зіркового многокутника будемо користуватись означенням [6], згідно з яким ядром зіркового многокутника є така множина точок, для якої усі відрізки, проведені з будь-якої точки цієї множини до будь-якої вершини зіркового многокутника, повністю належать цьому многокутнику. Тому перевірку многокутника на зірковість можна звести до пошуку ядра.

Твердження 1. Впорядкуємо усі ребра зіркового многокутника проти годинникової стрілки. Будь-яка точка належить ядру зіркового многокутника тоді, й тільки тоді, коли вона лежить зліва від усіх ребер цього многокутника.

Для розв’язання задачі пошуку ядра зіркового многокутника застосуємо персептрон Розенблата [9] (рис. 3).

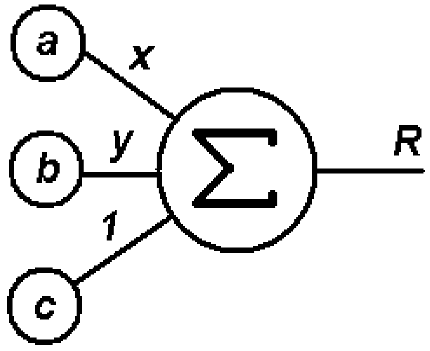


Рис.3. Персептрон Розенблата

На вхід персептрона подаємо коефіцієнти прямих, на яких лежать ребра. Причому, ці коефіцієнти є такими, що, якщо підставити будь-яку точку, яка лежить зліва від цієї прямої, у рівняння цієї прямої, то отримуємо невід’ємне число. Для точок, що лежать справа – не додатне. Навчаємо персептрон звичайним способом – методом градієнтного спуску. На вхід поступово подаємо коефіцієнти усіх прямих, на яких лежать ребра фігури, і поступово робимо це доти, доки не потрапимо до ядра зіркового многокутника або не досягнемо ліміту на кількість кроків. Якщо ліміт кроків досягнуто, а до ядра так і не потрапили, то вважаємо, що такий многокутник не є зірковим. Значення виходу персептрону рахуємо за формулою $R=ax+by+c$. Коефіцієнти x та y спочатку покладаємо рівними середньому арифметичному відповідних координат усіх вершин многокутника. На кожному кроці подаємо на персептрон усі прямі, на яких лежать ребра многокутника. Після кожної прямої, для якої точка (x, y) не лежить зліва, проводимо корегування x та y за формулами:

$$x_{n+1} = x_n - \gamma(ax+by+c)a, \quad y_{n+1} = y_n - \gamma(ax+by+c)b,$$

де γ – це швидкість навчання, яку можна змінювати.

3. Аналіз оцінки складності алгоритмів

Розв’язання поставленої задачі складається із 6 етапів, а тому проаналізуємо складність кожного етапу.

Етап 1. *Виокремлення вершин графу.* Складність процедури виокремлення вершин складає $O(n^2)$, де n – кількість відрізків, з яких складається вхідна фігура. Це пояснюється тим, що у найгіршому випадку для кожного відрізка потрібно переглянути усі для того, щоб з’ясувати, перетинаються вони чи ні.

Етап 2. *Видалення вершин першого ступеня та похідних вершин першого ступеня.* Складність процедури видалення вершин складає $O(m)$, де m – кількість вершин, що були виокремленні на попередньому кроці.

Отже, сумарна складність цих двох етапів буде наступною $O(n^2)+O(m)=O(n^2+m)$.

Етап 3. *Підготовка графу до аналізу методом ланцюгів.* Етап виконується за 5 проходів.

1. Орієнтація ребер та додання допоміжних. Алгоритм проходить усі ребра графу, а тому його складність $O(r)$, де r – кількість ребер у графі.
2. Сортування ребер за полярним кутом. Складність алгоритму $O(r\cdot\log r)$, бо у найгіршому випадку сортуються усі ребра.
3. Ініціалізація – присвоєння одиничної ваги усім ребрам($O(r)$).
4. Балансування ваг (проходом знизу вгору та згори вниз) ($O(r)$).
5. Таким чином, складність 3-го етапу складає $O(r)+O(r\cdot\log r)+ O(r)+O(r)+O(r)=O(4r+ r\cdot\log r)=O(r\cdot\log r)$.

Етап 4. *Алгоритм, що реалізує метод монотонних ланцюгів.* Основна процедура викликає рекурсивну функцію, яка здійснює пошук у глибину. Складність пошуку пропорційна кількості ребер у графі й складає $O(r)$.

Етап 5. *Алгоритм виокремлення фігур.* Етап складається з 2-х кроків:

1. *Виокремлення псевдо-фігур.* Виконується за наведеним раніш алгоритмом. Для обчислення складності позначимо: p – довжина найдовшого ланцюга. Для виокремлення фігур необхідно проглянути усі послідовні пари ланцюгів із метою знаходження точок перетину. Тому оцінка складності буде $O(p^3)$. Але кубічна оцінка не повинна лякати, бо як правило $p<<r$.

2. *Об’єднання псевдо-фігур.* Також виконується за наведеним раніш алгоритмом. Оцінка складності цього кроку напряму залежить від числа d – кількості псевдо-ребер у фігурах та f – кількості фігур. Так як для об’єднання псевдо-фігур потрібно виконати пошук псевдо-ребра та виконати його об’єднання з іншим, то складність кроку буде $O(d\cdot f\cdot p^2)$.

Етап 6. Алгоритм розпізнавання фігур та віднесення їх до певного класу. Етап складається з трьох кроків:

1. Визначення кількості кутів. Аналіз відбувається за наведеним вище алгоритмом. Складність $O(r_i)$, де r_i – кількість ребер у i-й фігурі.

2. Перевірка на опуклість. Виконується за лінійний час $O(r_i)$.

3. Якщо многокутник не є опуклим то він перевіряється на зірковість. Процедура ітеративна, а тому складність буде $O(Mr_i)$, де M – обмеження на число ітерацій.

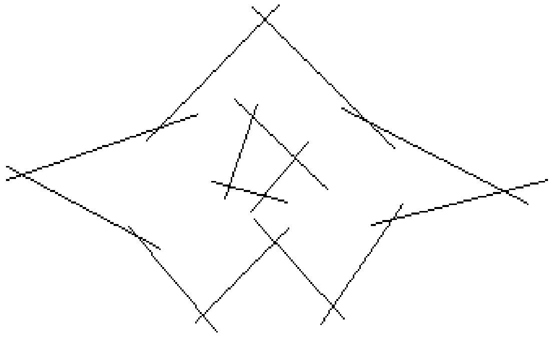
Отже, для повної реалізації розпізнавання геометричних образів на площині необхідно виконати усі наведені кроки. При цьому, складність кожного з них є функцією від результатів, отриманих на попередніх кроках.

4. Практична частина

На рис. 4 та рис. 5 подано приклад реалізації підходу, тобто, застосування методів обчислювальної геометрії та штучного інтелекту до задач комп’ютерного зору.

Постановка задачі:

1. На площині задано відрізки, що перетинаються



2. Необхідно визначити та розпізнати многокутники, що утворені перетином відрізків

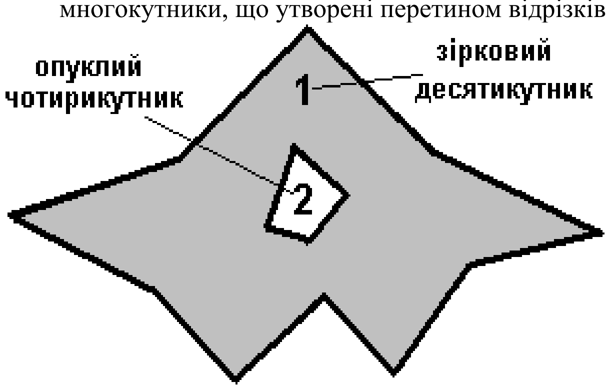
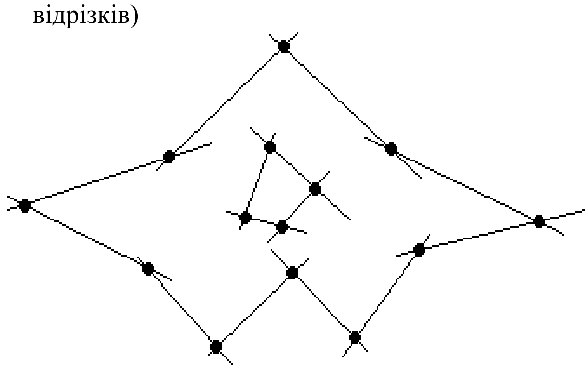


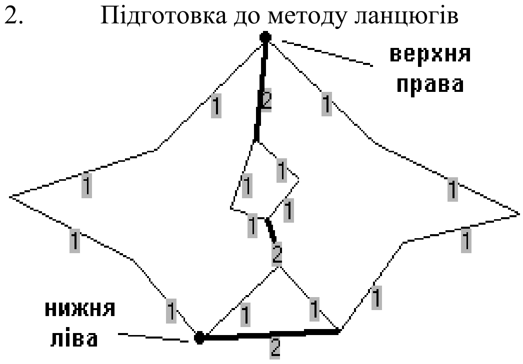
Рис.4. Постановка задачі

Алгоритм:

1. Виокремлення вершин (точок перетину відрізків)

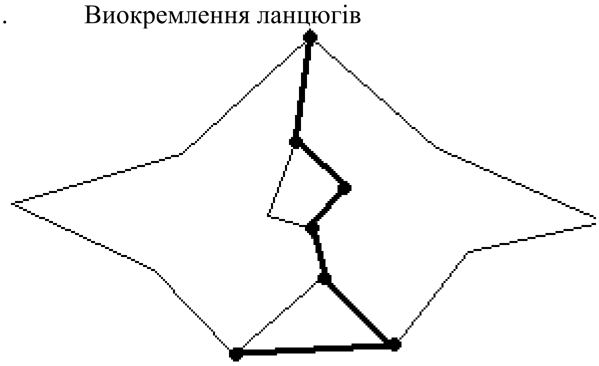


2. Підготовка до методу ланцюгів



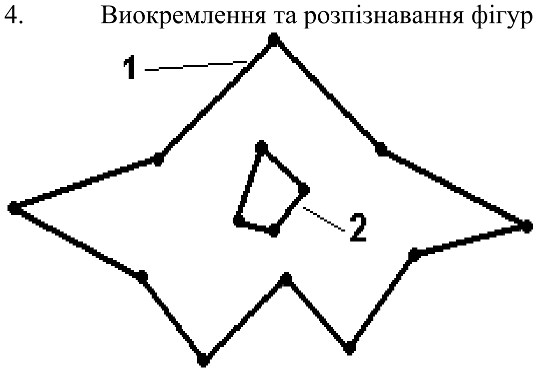
жирним – додаткові ребра

3. Виокремлення ланцюгів



жирним – один з ланцюгів

4. Виокремлення та розпізнавання фігур



1 – зірковий; **2 –** опуклий чотирикутник

Рис.5. Приклад алгоритму реалізації підходу

Висновки

Результати роботи мають практичні застосування у різних галузях штучного інтелекту, і зокрема, в задачах комп'ютерного зору. Усі зображення, як растрові, так і векторні, можна представити множиною відрізків, що перетинаються між собою. Апроксимація зображення прямими не тільки не змінює його змісту, але й навпаки,

дозволяє виокремити характерні риси зображення. Крім того, застосування векторного підходу цілком вирішує проблему масштабування образу до еталонного. Дуже часто класифікація образів здійснюється за критерієм: наскільки образ, що аналізується, схожий з тим чи іншим еталонним образом. Для растрових зображень це є не тривіальною задачею, але для векторних – це лише справа математичних перетворень. Окрім того, при переході від растрового зображення до векторного дуже часто майже повністю вдається уникнути шумів, що дуже спрощує обробку інформації. Також застосування векторного підходу буде дуже корисним для кодування інформації, отриманої з цифрових камер спостереження (чи інших схожих пристроїв). Ми отримаємо можливість зберігати та обробляти чисту інформацію, позбавлену шумів та зайвого фону. Запропонований підхід дозволяє суттєво економити місце для збереження архівних записів та проводити розпізнавання ситуацій у реальному часі. Причому, з'являється можливість реагувати не тільки на зміну ситуації у полі спостереження, а й розпізнавати саму ситуацію. Такий самий підхід можна застосувати й для стиснення цифрового відео, що зможе дозволити суттєво зекономити як місце на жорсткому диску, так і ресурси процесору. Це дасть змогу використовувати потужність відеокарт для відтворення відео.

Література

1. Дуда Р. Распознавание образов и анализ сцен / Дуда Р., Харт П. – М., 1976. – 507 с.
2. Верхаген К. Распознавание образов: состояние и перспективы / Верхаген К., Дейн Р., Грун Ф.– М. Радио и связь, 1985. – 104 с.
3. Васильев В.И. Распознающие системы / Васильев В.И. – Киев Наукова думка, 1983. – 422 с.
4. Горелик А.Л. Методы распознавания / Горелик А.Л., Скрипкин В.А. – М. Высшая школа, 2004. – 264 с.
5. Терещенко В.М. Персептронна модель прогнозування ринкових котирувань / В.М.Терещенко, Д.П.Григоренко // Вісник Київського університету: зб. наук. праць. –2009, Київ.– Вип. № 3.– С. 190–195.
6. Терещенко В.М. Принцип зводимості в задачах обчислювальної геометрії // Вісник Київського університету, серія: фізико-математичні науки, випуск № 2, 2009, ст. 149-154.
7. Preparata F. Computational Geometry / Preparata F., Lee D.T. – Berlin: Springer, 1985.– 357 c.
8. Lee D.T. Computational geometry: a survey / Lee D.T. – Berlin: Tucker, 1997.– 437 c.
9. Минский М. Персептроны / Минский М., Пейперт, 1971. – 262 с.
10. Уоссермен Ф. Нейрокомпьютерная техника / Уоссермен Ф. – М.: Мир, 1992. –506 с.
11. Горбань А.Н. Нейронные сети на персональном компьютере / Горбань А.Н., Россиев Д.А. – Новосибирск: Наука, 1996.– 276 с.
12. Ахо Х., Хопкрофт Дж. Построение и анализ вычислительных алгоритмов / Ахо Х., Хопкрофт Дж., Ульман Дж.М: Мир, 1979. – 536 с.
13. Роджерс Д. Алгоритмические основы машинной графики / Роджерс Д. – М.: «Мир», 1989. – 512 с.
14. Кнут Д. Искусство программирования для ЭВМ. Т. 1. Основные алгоритмы / Кнут Д. – М.: Мир, 1976. – 682 с.
15. Кнут Д. Искусство программирования для ЭВМ/Т. 3 Сортировка и поиск / Кнут Д. –С.-П.:Вильямс, 2000. – 824 с.
16. Фокс А. Вычислительная геометрия. Применение в проектировании и на производстве / Фокс А., Пратт М. – М.: Мир, 1982. – 304 с.
17. Ласло М. Вычислительная геометрия и компьютерная графика на С++ / Ласло М. – М.: Бином, 1997. –301 с.
18. Галахер Р. Метод конечного элемента. Основы / Галахер Р. – М.: Мир, 1974. – 428 с.
19. Фу К. Структурные методы в распознавании образов / Фу К. Мир, Москва 1977. – 319 с.
20. Информатика. Справочник. Под. Ред. Д.А.Поспелова. – Москва: Педагогика, 1996.
21. Гренадер У. Лекции по теории распознавания образов: В 3 т. / Гренадер У. – М., 1979. –1983.
22. De Berg M. Computational Geometry. Algorithms and Applications /Mark de Berg.–Berlin: Springer, 2000.–367 c.
23. Tereshchenko V. M., Zavershinskiy M. V. Some aspects of the search segment intersection triangles and spheres in R^3 // Applied geometry and graphics. – Kyiv, 2010. – No 85. – P. 192-198.

Literatura

1. Duda R. Raspoznavanye obrazov y analyz stsen / Duda R., Khart P. – M., 1976. – 507 s.
2. Verkhahen K. Raspoznavanye obrazov: sostoyanye y perspektyvы / Verkhahen K., Deyn R., Hrun F.– M. Radyo y svyaz', 1985. – 104 s.
3. Vasyl'ev V.Y. Raspoznayushchye systemы / Vasyl'ev V.Y. – Kyev Naukova dumka, 1983. – 422 s.
4. Horelyk A.L. Metodы raspoznavanyy / Horelyk A.L., Skrypkyn V.A – M. Vнsshaya shkola, 2004. – 264 s.
5. Tereshchenko V.M. Perseptronna model' prohnozuvannya rynkovykh kotyruvan' / V.M.Tereshchenko, D.P.Hryhorenko // Visnyk Kyyivs'koho universytetu: zb. nauk. prats'. –2009, Kyiv.– Vyp. 3.– S. 190–195.
6. Tereshchenko V.M. Pryntsyp zvodymosti v zadachakh obchyslyuval'noyi heometriyi // Visnyk Kyyivs'koho universytetu, seriya: fizyko-matematychni nauky, vypusk 2, 2009, st. 149-154.
7. Preparata F. Computational Geometry / Preparata F., Lee D.T. – Berlin: Springer, 1985.– 357 s.
8. Lee D.T. Computational geometry: a survey / Lee D.T. – Berlin: Tucker, 1997.– 437 s.
9. Mynskyy M. Perseptronы / Mynskyy M., Peypert, 1971. – 262 s.
10. Uossermen F. Neyrokomp'yuternaya tekhnyka / Uossermen F. – M.: Myr, 1992. –506 c.
11. Horban' A.N. Neyronnыe sety na personal'nom komp'yutere / Horban' A.N., Rossyev D.A. – Novosybyrsk: Nauka, 1996.– 276 s.
12. Akho Kh., Khopkroft Dzh. Postroenye y analyz vычyslytel'nыkh alhorytmov / Akho Kh., Khopkroft Dzh., Ul'man Dzh. Yzdatel'stvo: Myryu 1979. – 536 s.
13. Rodzhers D. Alhorytmycheskye osnovы mashynnoy hrafyky / Rodzhers D. – M.: «Myr», 1989. – 512 s.
14. Knut D. Yskusstvo prohrammyrovanyya dlya EVM. T. 1. Osnovnыe alhorytmы / Knut D. – M.: Myr, 1976. – 682 s.
15. Knut D. Yskusstvo prohrammyrovanyya dlya EVM/T. 3 Sortyrovka y poysk / Knut D. –S.-P.:Vyl'yams, 2000. – 824 s.
16. Foks A. Vychyslytel'naya heometryya. Prymenenye v proektyrovanyy y na proyzvodstve / Foks A., Pratt M. – M.: Myr, 1982. – 304 s.
17. Laslo M. Vychyslytel'naya heometryya y komp'yuternaya hrafyka na S++ / Laslo M. – M.: Bynom, 1997. –301 s.
18. Halakher R. Metod konechnoho эlementa. Osnovy / Halakher R. – M.: Myr, 1974. – 428 s.
19. Fu K. Strukturnыe metodы v raspoznavanyy obrazov / Fu K. Myr, Moskva 1977. – 319 s.
20. Ynformatyka. Spravochnyk. Pod. Red. D.A.Pospelova. – Moskva: Pedahohyka, 1996.
21. Hrenader U. Lektsyy po teoryy raspoznavanyya obrazov: V 3 t. / Hrenader U. – M., 1979. –1983.
22. De Berg M. Computational Geometry. Algorithms and Applications /Mark de Berg.–Berlin: Springer, 2000.–367 s.
23. Tereshchenko V. M., Zavershinskiy M. V. Some aspects of the search segment intersection triangles and spheres in R3 // Applied geometry and graphics. – Kyiv, 2010. – No 85. – P. 192-198.

RESUME

V. M. Tereshchenko , Y. V. Tereshchenko

One approach to recognizing geometric objects in computer vision problems

The paper is devoted to the use of one approach to addressing the various types of problems of pattern recognition. This approach is based on use of effective algorithms for computational geometry, and particular, the method of monotone chains. Also, in the paper we use Rosenblatt perceptron for recognition polygons.

The offered approach allows to develop a fast algorithm for solving the problem, and to identify the characteristics of the image and a wide range of applications in recognition problems, classification and graphic reproduction. Moreover, in the presented idea managed to combine the power of classical algorithms of computational geometry with the flexibility tools of artificial intelligence. We modified the classic chains method, applying it to recognition of geometric objects in the plane. The basic idea of this method is that we allocate on the given graph G (V, E) a linear order set of chains, which are monotonous relatively some direct, using an algorithm balancing weights. With these chains can easily allocate figures. Adjacent chains have at least two common points and therefore two adjacent chains form the definitely shape. During each viewing, we allocate among pairs of adjacent chains, all such chains that have common only a beginning and an end. Thus we can allocate polygons of respective classes. For example, to recognize of star polygon we use definition of star-polygon core. Then recognition is reduced to search star-polygon core. To search core star polygon we used Rosenblatt perceptron.

Надійшла до редакції 15.11.2016